Online Algorithms II

We saw the model last time, and the definition of competitive ratio:

$$c_{ratio} = \max_{\text{instance } I} \frac{\text{cost}(\text{algo}(I))}{\text{cost}(\text{OPT}(I))}$$

The performance of our algo:

The optimum cost in hindsight is an "offline" quantity.

**Ski Rental:**

A deterministic algo (Better-late-than-never), that buys m day B, if needed, achieves a

$$c_r \leq 2^{-1/2}.$$

**Proof:** If BLTN buys then pays $2B-1$

But their DPT $> B \Rightarrow c_r \leq \frac{2B-1}{B}$.

else season length $< B$

$\Rightarrow$ BLTN = OPT $\Rightarrow c_r \leq 1$ on these instances.

$\Rightarrow \max_I c_r(I) \leq \frac{2B-1}{B}.$

Note: No deterministic algo can be better than $2^{-1/2}$.

So Better-late-than-never is optimal (among deterministic algo).
Pf: Fix any deterministic algo \( A \).

Show that there exists an instance \( I \) such that
\[
\frac{\text{cost}(A(I))}{\text{cost}(\text{OPT}(I))} \geq \frac{2B-1}{B}.
\]

\( \text{Worry: Infinitely many deterministic algs?} \)

\( \text{Actually: Ski-Rental is nice because there are only finitely many "distinct" algs for us to consider.} \)

Any algo can be characterized by how many days it rents before it buys. Say this is \( T \).

\( \text{(So BLTN has } T = B-1). \)

Case \( T \geq B-1 \):

I instance = ski for \( T+1 \) days.

\[
\text{cost of } \text{OPT}(I) \leq B
\]

\[
\text{cost of } A(I) = T+B \geq (B-1)+B = 2B-1
\]

\[
\Rightarrow \text{ratio} \geq 2^{\frac{1}{B}}.
\]

Case \( T < B-1 \)

I = ski for \( T+1 \) days.

\[\text{OPT} = \text{rent all days, so } \text{cost}(\text{OPT}(I)) = T+1\]

\[\text{cost of } A(I) = T+B\]
So ratio = \( \frac{T+B}{T+1} = \frac{(T+1) + (B-1)}{(T+1)} = 1 + \left( \frac{B-1}{T+1} \right) \)

\[ \geq 1 + \frac{B-1}{B} = 2^{-\frac{1}{B}}. \]

So no matter what A we choose, \( \exists I \text{ s.t. } CR(I) \geq 2^{-\frac{1}{B}}. \)

Hence Bellu-Late-Thor-Neva is an optimal deterministic algo in the competitive analysis model.

Two things:

1. Suppose you allow randomization.

   You can buy on day 1 with some probability,
   day 2, etc.

   \( \Rightarrow \) can do better. Get \( \frac{e}{e-1} \) comp ratio (maybe tw)

2. Also if you have side information (say now instance drawn from some distribution based on prior predictions) then you may do better.

   But that is dependent on how good those predictions are.

   Competitive ratio is very robust, works for all inputs.
Next problem: Optimal Search.

You don't know $K$ where the keys are (positive or negative, or the value of $K$). So $\text{opt} = K$.

What's a good way to search?

How about: go to 1, go to -2, go to 3, -4, 5, 6, ...

Not good: $\text{opt} = K$

but also $= 1 + 3 + 5 + 7 + \ldots + (2K - 1)$

$= K^2$

$\Rightarrow c.K \geq K$.

Doubling search is much better

Go to 1, -2, 4, -8, 16, -32, \ldots \ldots \ldots until find keys.

What is distance traveled?

\[
\begin{align*}
K & \xrightarrow{2^1} K \\
& \xrightarrow{2^2} \text{K} \\
& \xrightarrow{2^4} 2^4 \\
& \xrightarrow{2^8} 2^8 \\
& \xrightarrow{2^{2^t}} 2^{2^t} \\
\end{align*}
\]
Suppose $k$ is negative and lies between $-2^{i-1}$ and $-2^{i+2}$ for some $i$.

$$\Rightarrow \text{distance traveled}$$

$$= 2(1 + 2 + 4 + 8 \cdots + 2^{i+r}) + |k|$$

$$= 2(2^{i+2} - 1) + |k| \leq 8.2^i + |k|$$

So $CR = \frac{|k| + 2^i \cdot q_0}{|k|} \geq 1 + 8 \cdot \frac{2^i}{|k|} \leq 9$.

**Theorem:** $CR \leq 9$ (and this is best possible for deterministic algs).

Not showing, requires a bit of work.

Again: can do better for randomized strategies.

Get $2e$ (which is optimal).

Extend to $L$ legs easily: comp ratio $O(L)$ possible.

Captures: problems where have to decide between strategies and there is a switching cost. Here switching costs are linear.
Problem III: List Update

Maintain a list, minimize the access cost. (see note)

Problem IV: Paging

Again, see notes.